Using and Interfacing Background Knowledge in Story Understanding

Nemecio R. Chavez, Jr.[1], Heather D. Pfeiffer[2], Roger T. Hartley[1]

[1]Department of Computer Science,
[2]Klipsch School of Electrical and Computer Engineering,
New Mexico State University
Box 30001, MSC CS/3-O, Las Cruces, NM, 88003-8001 USA
{nchavez,hdp,rth}@cs.nmsu.edu

Abstract. This paper details the use of background knowledge within a story understanding system. The story understanding system is based upon a multi-agent system (MAS). The MAS combines different forms of knowledge into a meaningful structure from which understanding can be demonstrated, i.e., question answering. The system uses two forms of knowledge to accomplish this task: 1) knowledge about objects in the world; and 2) prototypes. The latter represents higher-order knowledge or experience such as repeating a process and forms of thinking like abduction and deduction. The former stores knowledge about objects in the real world and their relationship to each other. The two forms are viewed as two distinct forms of data that might be used in our own understanding process, thus, they are treated separately by the system.

Keywords

Knowledge, knowledge bases, databases, communication, story understanding.

1 INTRODUCTION

Described are two forms of knowledge used to help understand a simple children's story. The bottom layer of knowledge provides *basic* knowledge needed for story understanding. The basic knowledge is information about the world that a 6 or 7 year old child might know and use. The story understanding system also provides an added layer of knowledge that will be referred to as *prototypes.* The prototype knowledge is similar to meta-data or meta-level knowledge and require concepts to be combined together therefore relaying a story as a group of *experience*s. The complete knowledge needed for understanding a story comes in two forms: 1) knowledge about objects in the world, and 2) higher-order knowledge or experience of objects and their relationships in the world. The story understanding system uses agents to combine this knowledge into a meaningful structure from which understanding can be demonstrated.

Consider the following text [1] taken from a children's book:

> Down at the very bottom of the pitcher there was a little water and the thirsty crow tried every way to reach it with her beak. But the pitcher was much too tall. The crow got thirstier and thirstier. At last she thought of a clever plan. One by one she dropped pebbles into the pitcher. Every pebble made the water rise a little higher. When the water reached the brim, the thirsty crow was able to drink with ease.

This text describes everyday objects and the relationships between those objects that most readers can understand very easily. In fact, a reader can demonstrate their understanding in a variety of ways. They can summarize the story, paraphrase it, retell it, or answer questions like the following:

- Why does the crow want the water?
- Why is the crow thirsty?
- How did the crow drop the pebbles into the water?
- What was the plan the crow thought of?
- Why did putting the pebbles in the pitcher make the water rise?

In the case of the latter, none of the questions can be answered from the text alone. They require the understanding of the text to be grounded in knowledge of the world; or rather they require in-depth understanding [16] to have occurred to be answered properly. The use of background knowledge and its importance to story understanding was recognized immediately [3].

Perhaps, one of the most studied avenues of story understanding is that of conceptual dependency [24] and scripts [25]. The idea is simple in that commonly encountered situations in our lives are broken up and stored as scripts. For example, each of us may have a Restaurant script in memory that provides us with all the information we have gained through our lives about restaurants. Thus, the meaning or understanding of a story becomes a series of scripts chained together in some form.

Script-based story understanding has had some success [26,2,8], but has been criticized for several points [13]. The most damaging being that scripts are based on data that appears tailor-made for the story at hand.

The purpose of this paper is to investigate the integration of knowledge used in a story understanding system and how using different levels of knowledge can create flexibility for storage and retrieval of information and which might shed light on some of the criticisms of story understanding systems (i.e., such as the data being tailor-made for the story or text at hand). Hobbs [9] uses a single intelligent agent to handle all the knowledge management, where we propose dividing the knowledge base into multiple agents (i.e., the basic and prototype knowledge). This paper also discusses the issue of interfacing between the understanding system and knowledge base .

2 KNOWLEDGE BASE

The knowledge base is broken up into two categories. In either case, the knowledge is at the level of a 6 or 7 years old child. The first describes knowledge about objects and their relationship to other objects. The second describes prototypes (higher-order knowledge or abstract knowledge) such as what it means to do something repeatedly, to drop something, or to like something, for example. The hope is that by separating the two, knowledge independent of the story understanding process can be isolated from knowledge the process is dependent on. Each category is discussed in the following subsections.

2.1 Knowledge about Objects

This part of the knowledge base is not tied to a particular domain. It is meant to be general knowledge about objects in the world that a child of 6 or 7 years old might know. As an example, consider the domain of the story from the introduction. A reader would need to know about crows, pitchers, pebbles, etc. to understand the text. In an English version of a knowledge base would need to be stored the following:

Water is a liquid.
Liquid is a thing.
All pitchers have a brim, bottom, and top.
Containers are things that hold things.
Pebbles are thing.
Crows are birds.
Birds are animals that have beaks.
Animals are living things.
Living things are things that need air, food, water, and shelter.

Note that while this knowledge is needed to understand the story, it is not necessarily specific to the story and can be used for other reasoning tasks. In fact, if some of this knowledge were missing, it is likely the reader would not fail to understand the whole story. They just would not understand the part that is missing or they might still be able to make correct inferences without it. For example, if pebbles were unknown, the reader still might be able to infer that putting them (whatever they are) in the pitcher caused the water level to rise and, thus, the crow was able to drink. Ideally, a story understanding system would be independent on this type of knowledge. Thus, it could be stored in any knowledge base with any format.

There are, essentially, three knowledge bases currently available about objects: Cyc, OpenMind, and ThoughtTreasure [6,17,27]. These knowledge bases (databases holding commonsense knowledge) are intended to capture the commonsense knowledge that we have of the world such as "A bird flies, unless it is a penguin." These knowledge bases are an attempt at building a single knowledge base of all known commonsense. Thus, researchers working at opposite ends of the world will be able to use them and expect the same level of consistency when commonsense knowledge is needed [9].

Although the ultimate goal is to capture the commonsense knowledge a human would have about the world, that goal is large. Thus, the focus of the knowledge has been on defining objects such as cars, liquids, animals, etc. and their relationships which include but are not limited to AKO (short for also known as), ISA (the inheritance relationship), PART-OF, etc. [16].

There is a major difference in philosophy in how the data is obtained within each base. OpenMind and ThoughtTreasure derive their knowledge by controlled interaction with the general public [16,17,27]. While on the other hand, Cyc comes from a corporation that employees people to engineer the knowledge [6]. Using engineered knowledge has the benefit that time was spent deciding on how the knowledge would be represented. However, using knowledge entered by the public has the benefit that the knowledge came from many independent minds. It is too soon to determine the overall impact of one philosophy over the other, but one would hope, regardless of which, eventually each database would cover the similar commonsense knowledge.

The story understanding system described here is not tied to any particular structure of data and, thus, is not tied to any particular commonsense database. However, for the initial prototype and because their source code was easily available and the ease with which they were to work with, the Conceptual Programming Environment (CPE) and the Knowledge Machine (KM) were used [19,5]. General everyday knowledge about the world was put into them and only knowledge about their query interface was held within the story understanding system. As an example of one of these systems, the CPE knowledge base is described in the following section.

CPE

The Conceptual Programming Environment, CPE, is a semantic network system built using conceptual structures [19]. Any knowledge base can be used with the story understanding system; however, this environment was chosen because 1) it has just been redesigned in order to be more flexible and to inter-operate with other systems [20], 2) the author is currently working on an internal module to allow reading and writing the new Common Logic CGIF format, and 3) it can incorporate knowledge from other large factual knowledge such as Cyc [6]. The knowledge is stored using the knowledge representation known as Conceptual Graphs, CGs, developed by John Sowa [28] using the ideas from C.S. Peirce's Existential Graphs [18]. There is also an interchange format, CGIF[1], which is available for exchanging conceptual knowledge.

The following is a translation of the English knowledge base above into CGIF notation.

```
;The first relations set up the type hierarchy for the concepts found in the knowledge base;
(GT [TypeLabel: ‘Entity’] [TypeLabel: ‘Thing’])
(GT [TypeLabel: ‘Thing’] [TypeLabel: ‘Liquid’])
(GT [TypeLabel: ‘Thing’] [TypeLabel: ‘Container’])
(GT [TypeLabel: ‘Thing’] [TypeLabel: ‘Pebble’])
(GT [TypeLabel: ‘Thing’] [TypeLabel: ‘LiveThing’])
(GT [TypeLabel: ‘Liquid’] [TypeLabel: ’Water’])
(GT [TypeLabel: ‘Container’] [TypeLabel: ‘Pitcher’])
(GT [TypeLabel: ‘LiveThing’][TypeLabel: ‘Animal’])
(GT [TypeLabel: ‘Animal’][TypeLabel: ‘Bird’])
(GT [TypeLabel: ‘Bird’][TypeLabel: ‘Crow’])
;The next set of Conceptual Graphs are definitional for giving structure to some of the concepts in
the type hierarchy just defined;
[Pitcher*p1:@every ;all pitchers;]
(ATTR ?p1 [Bottom])
(ATTR ?p1 [Top])
(ATTR ?p1 [Brim])
(HOLD [Container*c1] [Thing:@{}]
(PART [Bird] [Beak])
(NEED [PROPOSITION:
(BREATHE [LiveThing*lt1] [Air])
(EAT ?lt1 [Food])
(DRINK ?lt1 [Water])
(LIVEIN ?lt1 [Shelter])])
```

A Conceptual Graph is a bipartite, connected, directed graph where all nodes are partitioned into two disjoint sets, *concepts* and *conceptual relations*. The edges are directed arcs between concepts and conceptual relations [22]. Because these are true graphs as defined through graph theory, the last CGIF defined graph above, *LiveThing*, can be viewed in its display format in Figure 1.

Associated along with the concepts in the definitional graphs is where a concept lies in its context. The type hierarchy gives this contextual information. Conceptual graphs can represent not only definitional information, but also factual data that would be seen as the content data of a database. Lastly, content data can be assumed until it is known to be factual.

The reasoning algorithms within CPE use the projection and maximal join operations with the context of the type hierarchy over the definitional, factual, and assumption data graphs to build models within a query-answering system [19,22]. When the story understanding agent makes a query to the CPE system, then an instance of any appropriate CPE graph will be returned when there is a valid answer. The instance is a Java object that can be stored or more structural information can be retrieved from it.

The first object returned is the most probable result; then other probable objects will also be sent to the story understanding agent. However, CPE can do non-monotonic reasoning; that is, the objects

[1] The current benchmark copy of CGIF from the ICCS2001 workshop can be located at the web site: http://www.cs.nmsu.edu/~hdp/CGTools/cgstand/cgstandnmsu.html#Header_44.

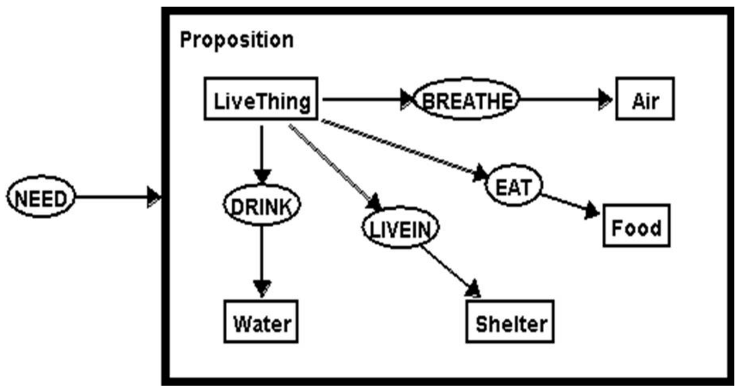

Fig. 1. Display format for a LiveThing Graph.

returned do not always hold with the concept that the hypotheses of any derived fact may be freely extended with additional assumptions. In fact, a true statement within the environment may not continue to be true when new added axioms or facts are introduced into the environment [14]. In this way, as knowledge is added to the knowledge base, a query from the story understanding agent may get different results or even just the same object in a different order of probability. In CPE all the knowledge is totally independent of the story understanding system and can be used by any application, or even enhanced by other running systems to give more correct information.

2.2 Prototype Knowledge

Prototypes represent abstract or general knowledge. For example, all users have some notion of what it means to repeat or to drop something. In fact, there is a notion of the meaning based on context. In the context of the everyday world, repeating something may simply mean to do something over and over, such as asking a question more than once. Thus, if we ask someone for a quarter repeatedly, the meaning might be akin to the following:

Do you have a quarter I can borrow?
Do you have a quarter I can borrow?
.
.
.

However, in the context of a programming language, it might be said that repeating something means to perform those statements within a loop. It would not be unusual for a programming instructor to say, "Then we repeat these statements for the length of the array." Our understanding of this statement might be in terms of a specific looping structure, or perhaps a generic one akin to the one below.

```
Loop from 1 to length of array
// repeat statements
End loop
```

Aside from the obvious, each of these understandings have a different meaning because they could potential conjure up new ideas or things that we know about what was said. This is important in that we might be able to answer a wider variety of questions about what we read.

It might seem like prototypes are simply templates that we fill in with other prototypes and knowledge of the world. However, they are much more than that because they not only help expand or constrain our understanding (such as with repeating something like above), but they help guide our thinking or act as a train of thought for our thinking. For example, when we read something, we often times know in advance the genre of the material. If we were reading a comic strip, we might not take it as seriously as we would a scientific paper. A prototype could be used for the genre and, thus, guide our thinking as something is being comprehended.

Other examples of these prototypes include scripts. Recall that scripts provide a general recipe for doing everyday things in our lives. Going to a restaurant, scheduling a trip, purchasing tickets, are all general guides of what is involved in when we normal do these things. Thus, any script is a prototype.

Prototype knowledge seems difficult to store in a traditional knowledge base (albeit a concept denoting the knowledge could be easily stored), but could certainly benefit from some of the reasoning capabilities most knowledge bases provide. Recently, there was the suggestion that scripts could make use of being stored in a hierarchical fashion [10]. Thus, object-oriented techniques show some promise in working with prototypes.

3 THE STORY UNDERSTANDING SYSTEM

The story understanding system described here is in the process of being developed. A prototype is currently working and the next evolution is currently being built. This system is an attempt at satisfying the following metrics:

- It should be capable of performing in-depth story understanding [16];
- Revisions to the system or knowledge should be isolated to manageable, well-defined areas or domains [16,23];
- Knowledge, in general, should be independent of the text at hand [12];
- It should have a high-level of elaboration tolerance [15];
- Semantic problems or errors should be detectable and if possible recoverable [21].

These metrics were built from the goals and criticism of previous work (suggestions or actual systems) [4]. The system tries to satisfy these metrics in a variety of ways.

To perform in-depth story understanding, the system uses knowledge about the world and prototypes to build understanding that goes beyond the given text. To help with the system or knowledge being isolated to manageable, well-defined areas the system uses agents based on each proposition to help bind the knowledge about objects with the prototype knowledge. This system is also broken up into five components based upon the five levels of representation psychologists generally believe we use when comprehending written text [7], which helps make the system manageable and provides a method for dealing with semantic errors between the components. The system works one sentence at a time, so it tries to make adjustments to understanding already in place instead of discarding what is already known. Thus, the system is elaboration tolerant [15].

To simplify things, the system takes input sentences at the text-base level. That is, sentences that have their meaning preserved, but which have lost their exact wording or syntax. These sentences would be composed of propositions. A sentence at this level would map directly to known propositions in memory. For example, the following sentence:

One by one she dropped pebbles into the pitcher.

Would be converted to the following form ahead of time:

```
repeat[dropped[AGENT=she, OBJECT=pebble,
               ACTION=into, TARGET=pitcher]]
```

Pebble and pitcher would need to exist in the object knowledge base and the rest would need to exist in the prototype knowledge base. In fact, it can be assumed they exist because to convert a sentence to its text-base form would require knowledge of objects and prototypes existing in the knowledge base.

3.1 The Understanding Process

There is a special section of prototype memory where a multi-agent system lives. The agents are the binding force between the prototypes and the objects. Processing or understanding occurs through the agents. A story is processed sentence by sentence. As each proposition is encountered agents are fired. Each agent knows what prototypes and what objects are necessary to build the understanding of the given proposition. This processing is recursive, thus, implying the meaning of a proposition is defined in terms of other propositions as well as itself.

Another way to think of the agents is that they are merely data themselves. This algorithm is processing the propositions, but it is processing them in terms of the agents. Thus, the process forces the form or the structure of the data, or rather agents, to be shared.

A simple pseudo-code version of the top-level understanding agent (i.e., algorithm) is given below:

```
understanding = null
while there are propositions
    fire agent for current proposition
    process sub-propositions for given proposition
    update understanding
    get next proposition
end
```

Once processing of the text is finished, a structure containing a binding of prototypes and knowledge about objects is created. The prototypes of course handle the high-level knowledge while the objects simply handle basic properties about themselves and their relationship to other objects. This finished structure can be used for questions answering, summarizing, outlining, etc.

It should be noted that this processing may be recursive because the meaning of a proposition may be defined in terms of other propositions as well as itself. Semantic error detection would also be part of the processing. For example, consider the following sentence:

The water rose before each pebble was placed in the pitcher.

This would cause an error because the water would only rise after each pebble was placed in the pitcher.

3.2 Interfacing between Story System and Knowledge Base

Part of the work in designing this story understanding system, is to consider what would be needed to define a generic interface between the system and knowledge bases. This is a very difficult task because most knowledge bases provide access to their knowledge via a query language specific to the knowledge base. Thus, a generic interface between an application and knowledge base would have to define some mapping between queries going through the interface and those going to the knowledge base. The queries from the generic interface would stay consistent across all applications interfacing with it. However, the query from the interface to the knowledge base would be specific to whatever knowledge base is being used. This provides a challenging and time-consuming task.

Another interesting aspect is that story understanding systems would benefit from certain queries being intrinsic to the generic interface. For example, when converting a sentence from English to the text-base level (i.e., to propositional form), knowledge of what concepts and related concepts are stored in the knowledge base would be helpful. Any English sentence would need to be converted to propositional form at some point. Thus, verifying or having access to known concepts is essential to this process. Likewise, knowing aliases or related concepts and when they should be used would beneficial.

Some very preliminary work has been done with respect to this idea [21]. This work only considered the problem of automatically translating knowledge between knowledge bases so the same knowledge was contained with in each. It is limited to very small well-defined domains because as the size of the domain grows it becomes more and more difficult to ensure all consequences of the knowledge base or maintained.

4 CONCLUSIONS

The CPE environment and story understanding system described in this paper provide only a small step towards an ideal system. This collaboration shows that communication between a story understanding system and a knowledge base can be done transparently. This also allows the understanding system to use multiple sources and categories of knowledge, which improves understanding, and allows the knowledge base and system to be loosely coupled [11]. Thus, reducing or potentially eliminating the amount of hand-tailored data [12]. Also, knowledge does not have to be transferred between the different knowledge systems to give current and up to date content data, but can continue to be stored within the current system and just shared as needed,

Knowledge like prototypes could benefit from being stored in a lower level knowledge base as a single intelligent agent, but would not be as flexible as leveling the knowledge and having multiple agents. By using a standardized interface to the low level knowledge base, perhaps, the story understanding agents could be achieved without any specific knowledge dependencies. The story understanding agents could, thus, only query for object information when needed for a prototype they are working on. This allows the basic knowledge base to hold more factual information (such as they loaded from a standard knowledge base like Cyc) and just "service" the understanding agents.

References

[1] Aesop, *Aesop's fables*, Doubleday & Company, Garden City Books (1954).

[2] Alvarado, S.J., *Understanding Editorial Text: A Computer Model for Argument Comprehension*, Kluwer Academic, Boston, Mass (1990).

[3] Charniak, E., "Toward a model of children's story comprehension", *AI Laboratory Technical Report 266, Artificial Intelligence Laboratory*, Massachusetts Institute of Technology (1972).

[4] Chavez, Jr., N.R. and Hartley, R.T., "The Role of Object-Oriented Techniques and Multi-Agents in Story Understanding", *Proceedings of the International Conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS 2005)*, Waltham, Mass (2005).

[5] Clark, P. and Porter, B., "Using Access Paths to Guide Inference with Conceptual Graphs", in D. Lukose, H. Delugach, M. Keeler, L. Searle, J. Sowa (Eds): *Proc Int Conf on Conceptual Structures - ICCS'97 (Lecture Notes in AI vol 1257)*, pages 521-535, Berlin:Springer (1997).

[6] Cycorp, OpenCyc, http://www.opencyc.org.

[7] Graesser, A.C., Millis, K.K., and Zwaan, R.A., "Discourse Comprehension", *Annual Review of Psychology,* Volume 48, pages 163-190 (1997).

[8] Grishman, R., and Sundheim, B., "Message Understanding Conference - 6: A brief history", *6th International Conference on Computational Linguistics*, pages 466-471 (1996).

[9] Hobbs, J.R., and Gordon, A.S., "Encoding Knowledge of Commonsense Psychology", *7th International Symposium on Logical Formalizations of Commonsense Reasoning*, Corfu, Greece, pages 107-114 (2005).

[10] Kalantzis, G.D., "SOPHIA: An Integrated Model for Story Understanding", *Proceedings of the 3rd Annual CLUK Research Colloquium*, Brighton (2000).

[11] Keeler, M. and Pfeiffer, H.D., "Collaboratory testbed partnerships as a knowledge capture challenge", in P. Clark and G. Schreiber, (Eds): *Proceedings of the Third International Conference on Knowledge Capture*, pages 203-204, KCAP'05, ACM Press (2005).

[12] Keeler, M.A. and Pfeiffer, H.D., "Building a Pragmatic Methodology for KR Tool Research and Development", in H. Scharfe, P. Hitzler, and P. Ohrstrom, (Eds): *Conceptual Structures: Inspiration and Application, LNAI, Proceedings of ICCS 2006*, Aalborg, Denmark, pp. 314-330, Berlin:Springer (2006).

[13] Mallery, J.C., "Thinking About Foreign Policy: Finding an Appropriate Role for Artificially Intelligent Computers", Cambridge: Master's Thesis, M.I.T. Political Science Department, (1988).

[14] Marek,W. and M. Truszczynski , *Nonmonotonic Logics: Context-Dependent Reasoning.* Springer Verlag, (1993).

[15] McCarthy, J.,"Elaboration tolerance", *The 1998 Symposium on Logical Formalizations of Commonsense Reasoning*, London (1998).

[16] Mueller, E.T., "Story understanding", *Encyclopedia of Cognitive Science*, London: Nature Publishing Group (2002).

[17] Mueller, E.T., ThoughtTreasure, http://www.signiform.com/tt/htm/tt.htm.

[18] Peirce, C.S.. Manuscripts on existential graphs. *Peirce*, 4:320–410, (1960).

[19] Pfeiffer, H.D., and Hartley, R.T., "Temporal, spatial, and constraint handling in the conceptual programming environment, CP", *Journal for Experimental and Theoretical AI*, 4(2):167–182, (1992).

[20] Pfeiffer, H.D., "An Exportable CGIF Module from the CP Environment: A Pragmatic Approach", in K.E. Wolff, H.D. Pfeiffer, and H.S. Delugach (Eds.): *Lecture Notes in Artificial Intelligence, Vol. 3127,* pp. 319-332, Springer-Verlag, (2004).

[21] Pfeiffer, H.D, Chavez, Jr., N.R., and Hartley, R.T," A Generic Interface for Communication between Story Understanding Systems and Knowledge Bases", *Richard Tapia Celebration of Diversity in Computing Conference*, (2005).

[22] Pfeiffer, H.D. and Hartley, R.T., "A Comparison of Different Conceptual Structures Projection Algorithms", in *the 15th International Conference on Conceptual Structures,* Sheffield Hallam University, Sheffield, UK, July (2007).

[23] Riloff, E., "Information Extraction as a Stepping Stone toward Story Understanding", *Computational Models of Reading and Understanding*, MIT Press, 435-460, (1999).

[24] Schank, R.C., *Conceptual Information Processing*, American Elsevier, (1975).

[25] Schank, R.C., and Abelson, R., *Scripts, Plans, Goals and Understanding*, Lawrence Erlbaum, Hillsdale, N.J, (1977).

[26] Schank, R.C., and Riesbeck, C.K., *Inside Computer Understanding*, Lawrence Erlbaum, Hillsadale, N.J., (1981).

[27] Singh, P., OpenMind,.http://www.openmind.org.

[28] Sowa, J.F., *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley, Reading, Mass, (1984).